Shift & 2D Rotation Invariant Sparse Coding for Multivariate Signals

Quentin Barthélemy, Anthony Larue, Aurélien Mayoue, David Mercier, Jerome I. Mars

Shift & 2D Rotation Invariant Sparse Coding for Multivariate Signals

Quentin Barthélemy, Anthony Larue, Aurélien Mayoue, David Mercier, and Jérôme I. Mars, *Member, IEEE*

***Abstract*—Classical dictionary learning algorithms (DLA) allow unicomponent signals to be processed. Due to our interest in two-dimensional (2D) motion signals, we wanted to mix the two components to provide rotation invariance. So, multicomponent frameworks are examined here. In contrast to the well-known multichannel framework, a multivariate framework is first introduced as a tool to easily solve our problem and to preserve the data structure. Within this multivariate framework, we then present sparse coding methods: multivariate orthogonal matching pursuit (M-OMP), which provides sparse approximation for multivariate signals, and multivariate DLA (M-DLA), which empirically learns the characteristic patterns (or features) that are associated to a multivariate signals set, and combines shift-invariance and online learning. Once the multivariate dictionary is learned, any signal of this considered set can be approximated sparsely. This multivariate framework is introduced to simply present the 2D rotation invariant (2DRI) case. By studying 2D motions that are acquired in bivariate real signals, we want the decompositions to be independent of the orientation of the movement execution in the 2D space. The methods are thus specified for the 2DRI case to be robust to any rotation: 2DRI-OMP and 2DRI-DLA. Shift and rotation invariant cases induce a compact learned dictionary and provide robust decomposition. As validation, our methods are applied to 2D handwritten data to extract the elementary features of this signals set, and to provide rotation invariant decomposition.**

***Index Terms*—Dictionary learning algorithm, handwritten data, multichannel, multivariate, online learning, orthogonal matching pursuit, rotation invariant, shift-invariant, sparse coding, trajectory characters.**

I. Introduction

In the signal processing and machine-learning communities, sparsity is a very interesting property that is used more and more in several contexts. It is usually employed as a criterion in a transformed domain for compression, compress sensing, denoising, demoisaicing, etc. [1]. As we will consider, sparsity can also be used as a feature extraction method, to make emerge from data the elements that contain relevant information. In our application, we focus on the extraction of primitives from the motion signals of handwriting.

Manuscript received June 14, 2011; revised October 18, 2011 and December 20, 2011; accepted December 20, 2011. Date of publication January 06, 2012; date of current version March 06, 2012. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Alfred Hanssen.

Q. Barthélemy, A. Larue, A. Mayoue, and D. Mercier are with CEA-LIST, Data Analysis Tools Laboratory, 91191 Gif-sur-Yvette Cedex, France (e-mail: quentin.barthelemy@cea.fr).

J. I. Mars is with GIPSA-Lab, 38402 Saint Martin d'Hères, France (e-mail: jerome.mars@gipsa-lab.grenoble-inp.fr).

Digital Object Identifier 10.1109/TSP.2012.2183129

To process signals in a Hilbert space, we define the matrix inner product as $\langle A, B\rangle = \mathrm{trace}(B^H A)$, with $(\cdot)^H$ representing the conjugate transpose operator. Its associated Frobenius norm is represented as $\|\cdot\|$. Considering a signal $y \in \mathbb{C}^N$ that is composed of N samples and a dictionary $\Phi \in \mathbb{C}^{N\times M}$ composed of M atoms $\{\phi_m\}_{m=1}^M$, the decomposition of the signal y is carried out on the dictionary Φ such that

$$y = \Phi x + \epsilon \tag{1}$$

assuming $x \in \mathbb{C}^M$, the coding coefficients, and $\epsilon \in \mathbb{C}^N$, the residual error. The approximation of y is $\hat{y} = \Phi x$. The dictionary is generally normed, which means that its columns (atoms) are normed, so that the coefficients x reflect the energy of each atom present in the signal. Moreover, the dictionary is said redundant (or overcomplete) when $M > N$: the linear system of (1) is thus underdetermined and has multiple possible solutions. The introduction of constraints, such as positivity, sparsity or others, allows the solution to be regularized. In particularly, the decomposition under a sparsity constraint is formalized by

$$\min_x \|x\|_0 \text{ s.t. } \|y - \Phi x\|^2 \leq C_0 \qquad (P_0)$$

where C_0 is a constant, and $\|x\|_0$ the ℓ_0 pseudo-norm is defined as the cardinal of the x support[1]. The formulation of (P_0) includes a term of sparsification to obtain the sparsest vector x and a data-fitting term.

To obtain the sparsest solution for (P_0), let us imagine a dictionary Φ that contains all possible patterns. This allows any signal to be approximated sparsely, although this would be too huge to store and the coefficients estimation would be intractable. Therefore, we have to make a choice about the dictionary used, with there being three main possibilities.

First, we can choose among classical dictionaries, such as Fourier, wavelets [2], curvelets [3], etc. If these generic dictionaries allow fast transforms, their morphologies deeply influence the analysis. Wavelets are well adapted for studying textures, curvelets for edges, etc., each dictionary being dedicated to particular morphological features. So, to choose the *ad hoc* dictionary correctly, we must have an *a priori* about the expected patterns.

Second, several of these dictionaries can be concatenated, as this allows the different components to be separated, each being sparse on its dedicated subdictionary [4], [5]. If it is more flexible, we always need to have an *a priori* of the expected patterns.

Third, we let the data choose their optimal dictionary themselves. Data-driven dictionary learning allows *sparse coding*: elementary patterns that are characteristic of the dataset are learned empirically to be the optimal dictionary that jointly

[1]The support of x is $\mathrm{support}(x) = \{m \in \mathbb{N}_M : x_m \neq 0\}$.

gives sparse approximations for all of the signals of this set [6]–[8]. The atoms obtained do not belong to classical dictionaries: they are appropriate to the considered application. Indeed, for practical applications, learned dictionaries have better results than classical ones [9], [10]. The fields of applications comprise image [6]–[8], [11], audio [12]–[14], video [15], audio–visual [16], and electrocardiogram [17], [18].

For multicomponent signals, we wish to be able to sparsely code them. For this purpose, the existing methods concerning sparse approximation and dictionary learning need to be adapted to the multivariate case. Moreover, in studying 2D motions acquired in bivariate real signals, we want the decompositions to be independent of the orientation of the movement execution in the 2D space. The methods are thus specified for the 2D rotation invariant (2DRI) case so that they are robust to any rotation.

Here, we present the existing sparse approximation and dictionary learning algorithms in Section II, and we look at the multivariate and shift-invariant cases in Section III. We then present multivariate orthogonal matching pursuit (M-OMP) in Section IV, and the multivariate dictionary learning algorithm (M-DLA) in Section V. To process bivariate real signals, these algorithms are specified for the 2DRI case in Section VI. For their validation, the proposed methods are applied to handwritten characters in Section VII for several experiments. We thus aim at learning an adapted dictionary that provides rotation invariant sparse coding for these motion signals. Our methods are finally compared to classical dictionaries and to existing learning algorithms in Section VIII.

II. State of the Art

In this section, the state of the art is given for sparse approximation algorithms, and then for DLAs. These are expressed for unicomponent signals.

A. Sparse Approximation Algorithms

In general, finding the sparsest solution of the coding problem (P_0) is NP-hard [19]. One way to overcome this difficulty is to simplify (P_0) in a subproblem:

$$\min_x \|y - \Phi x\|^2 \text{ s.t. } \|x\|_0 \leq K \qquad (P_0')$$

with $K \ll M$, a constant. Pursuit algorithms [20] tackle (P_0') sequentially by increasing K iteratively, although this optimization is nonconvex: the solution obtained can be a local minimum. Among the multiple ℓ_0-pursuit algorithms, the following examples are useful here: the famous matching pursuit (MP) [21] and its orthogonal version, the OMP [22]. Their solutions are suboptimal because the support recovery is not guaranteed, especially for a high dictionary coherence[2] μ_Φ. Nevertheless, they are fast when we search very few coefficients [23].

Another way consists of relaxing the sparsification term of (P_0) from a ℓ_0 norm to a ℓ_1 norm. The resulting problem is called basis pursuit denoising [4]:

$$\min_x \|x\|_1 \text{ s.t. } \|y - \Phi x\|^2 \leq C_1 \qquad (P_1)$$

with C_1 a constant. (P_1) is a convex optimization problem with a single minimum, which is the advantage with respect to ℓ_0-Pursuit algorithms. Under some strict conditions [1], the solution obtained is the sparsest one. Different algorithms for solving this problem are given in [20], such as methods based on basis pursuit denoising [4], homotopy [24], iterative thresholding [25], etc. However, a high coherence μ_Φ does not ensure that these algorithms recover the optimal x support [1], and if this is the case, the convergence can be slow.

[2]The coherence of the normed dictionary Φ is $\mu_\Phi = \max_{i \neq j} |\langle \phi_i, \phi_j \rangle|$.

B. Dictionary Learning Algorithms

The aim of DLAs is to empirically learn a dictionary Φ adapted to the signals set that we want to sparsely code [26]. We have a training set $Y = \{y_p\}_{p=1}^P$, which is representative of all of the signals studied. In dictionary learning, interesting patterns of the training set are iteratively selected and updated. Most of the learning algorithms alternate between two steps:

1) the dictionary Φ is fixed, and coefficients x are obtained by sparse approximation;
2) x is fixed, and Φ is updated by gradient descent.

Old versions of these DLAs used gradient methods to compute the coefficients x [6], while new versions use sparse approximation algorithms [7], [11]–[14], [27]. Based on the same principle, the method of optimal directions (MOD) [17] updates the dictionary with the pseudo-inverse. This method is generalized in [18] under the name of iterative least-squares DLA (ILS-DLA). There are also methods that do not use this principle of alternative steps. K-SVD [8] is a simultaneous learning algorithm, where at the 2nd step the x support is kept: Φ and x are updated simultaneously by SVD.

At the end of all these learning algorithms, the dictionary that is learned jointly provides sparse approximations of all of the signals of the training set: it reflects sparse coding.

III. Multivariate and Shift-Invariant Cases

In this section, we consider more particularly the multivariate and the shift-invariant cases. Moreover, the link between the classical (unicomponent) and the multivariate framework is discussed.

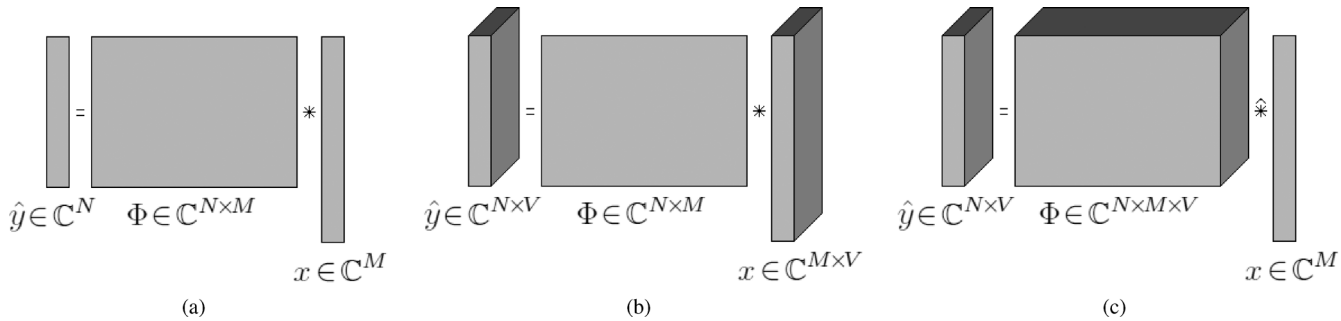
A. Multivariate Case

Up to this point, a unicomponent signal $y \in \mathbb{C}^N$ has been examined and its classical framework approximation is illustrated in Fig. 1(a). In the multicomponent case, the signal studied becomes $y \in \mathbb{C}^{N \times V}$, with V denoting the number of components. Two problems can be considered, which depending on the natures of Φ and x:

- $\Phi \in \mathbb{C}^{N \times M}$ unicomponent and $x \in \mathbb{C}^{M \times V}$ multicomponent, the well-known *multichannel* framework [Fig. 1(b)];
- $\Phi \in \mathbb{C}^{N \times M \times V}$ multicomponent and $x \in \mathbb{C}^M$ unicomponent, the *multivariate* framework [Fig. 1(c)], which considers Φx as an element-wise product along the dimension M.

The difference between the multichannel and multivariate frameworks is the approximation model, and we will detail this for both frameworks.

Multichannel sparse approximation [28]–[31] is also known as simultaneous sparse approximation (SSA) [32]–[36], sparse approximation for multiple measurement vector (MMV) [37], [38], joint sparsity [39] and multiple sparse approximation [40]. In this framework, all of the components have the same dictionary and each component has its own coding coefficient. This means that all components are described by the same profiles



(a) (b) (c)

Fig. 1. Decomposition with (a) classical, (b) multichannel, and (c) multivariate frameworks. In (c), $\hat{*}$ is considered as an element-wise product along the dimension M.

(atoms), although with different energies: each profile is linearly mixed in the different channels. This framework is also known as a blind source separation problem.

The multivariate framework can be considered as the inverse framework: all of the components have the same coding coefficient, and thus the multivariate signal y is approximated sparsely as the sum of a few multivariate atoms ϕ_m. Data that come from different physical quantities, that have dissimilar characteristic profiles, can be aggregated in the different components of the multivariate kernels: they must only be homogeneous in their dimensionalities. To our knowledge, this framework has been considered only in [41] for an MP algorithm, although with a particular dictionary template that included a mixing matrix. In the present study, we focus mainly on this multivariate framework, with Φ multivariate and normed (i.e., each multivariate atom is normed, such that $\|\phi_m\| = 1$).

In this paragraph, we consider DLAs that deal with multicomponent signals. Based on the multichannel framework, the dictionary learning presented in [42] uses a multichannel MP for sparse approximation and the update of K-SVD. We note that the two channels considered are then updated alternatively. In bimodal learning with audio–visual data [16], each modality (audio/video) has its own dictionary and its own coefficient for the approximation, and the two modalities are updated simultaneously. We also mention [43] based on the multichannel framework: they used multiplicative updates for ensuring the nonnegativity of parameters.

B. The Shift-Invariant Case

In the shift-invariant case, we want to sparsely code the signal y as a sum of a few short structures, known as kernels, that are characterized independently of their positions. This model is usually applied to time series data, and it avoids block effects in the analysis of largely periodic signals and provides a compact kernel dictionary [12], [13].

The L shiftable kernels (or generating functions) of the compact dictionary Ψ are replicated at all of the positions, to provide the M atoms (or basis functions) of the dictionary Φ. The N samples of the signal y, the residue ϵ, and the atoms ϕ_m are indexed[3] by t. The kernels $\{\psi_l\}_{l=1}^L$ can have different lengths. The kernel $\psi_l(t)$ is shifted in the τ samples to generate the atom $\psi_l(t-\tau)$: zero padding is carried out to have N samples. The subset σ_l collects the active translations τ of the kernel $\psi_l(t)$. For the few kernels that generate all of the atoms, (1) becomes

$$\begin{aligned} y(t) &= \sum_{m=1}^{M} x_m \phi_m(t) + \epsilon(t) \\ &= \sum_{l=1}^{L} \sum_{\tau \in \sigma_l} x_{l,\tau} \psi_l(t-\tau) + \epsilon(t). \end{aligned} \tag{2}$$

Due to shift-invariance, Φ is the concatenation of L Toeplitz matrices [14], and is L times overcomplete. In this case, the dictionary is said convolutional. As a result, in the present study, the multivariate signal y is approximated as a weighted sum of a few shiftable multivariate kernels ψ_l.

Some DLAs are extended to the shift-invariant case. Here all of the active translations of a considered kernel are taken into account during the update step [12]–[15], [44], [45]. Furthermore, some of them are modified, to deal with the disturbances introduced by the overlapping of the selected atoms, such as extensions of K-SVD [46], [47] and ILS-DLA (with a shift factor of 1) [48].

C. Remarks on the Multivariate Framework

Usually, the multivariate framework is approached using vectorized signals. The multicomponent signal is vertically vectorized from $y \in \mathbb{C}^{N \times V}$ to $y \in \mathbb{C}^{NV \times 1}$, and the dictionary from $\Phi \in \mathbb{C}^{N \times M \times V}$ to $\Phi \in \mathbb{C}^{NV \times M}$. After applying the classical OMP, the processing is equivalent to the multivariate one. In this paragraph, we explore the advantages of using the multivariate framework, rather than the classical (unicomponent) one.

In our case, the different components are acquired simultaneously, and the multivariate framework allows this temporal structure of the acquired data to be keep. Moreover, these components can be very heterogeneous physical quantities. Vectorizing components causes a loss of physical meaning for signals and for dictionary kernels, and more particularly when the components have dissimilar profiles. We prefer to consider the data studied as being multicomponent: the signals and the dictionary have several simultaneous components, as illustrated in the following figures. For the algorithmic implementation, in the shift-invariant case the multivariate framework is easier to implement and has a lower complexity than the classical one with vectorized data (see Appendix A).

Moreover, the multivariate framework sheds new light on the topic when multicomponent signals are being processed (see Section IV-B). Furthermore, presented in this way, the

[3]Remark that $a(t)$ and $a(t-t_0)$ do not represent samples, but the signal a and its translation of t_0 samples.

2DRI case is viewed as a simple specification of the multivariate framework, mostly involving the selection step (see Section VI). The rotation mixes the two components which are acquired simultaneously but which were independent previously, that is easy to see with multivariate signals as opposed to vectorized ones.

Consequently, the multivariate framework is principally introduced for the clearness of the explanations and for the ease of algorithmic implementation. Thus, we are going to present multivariate methods that existed under another less-adapted formalism, and were introduced in [49] and [50]: multivariate OMP and multivariate DLA.

IV. Multivariate Orthogonal Matching Pursuit

In the present study, sparse approximation can be achieved by any algorithm that can overcome the high coherence that is due to the shift-invariant case. For real-time applications, OMP is chosen because of its tradeoff between speed and performance [23]. In this section, OMP and M-OMP are explained step by step.

A. OMP Presentation

As introduced in [22], OMP is presented here for the unicomponent case and with complex signals. Given a redundant dictionary, OMP produces a sparse approximation of a signal y (Algorithm 1). It solves the least squares problem (P_0') on an iteratively selected subspace.

After initialization (step 1), at the current iteration k, OMP selects the atom that produces the absolute strongest decrease in the mean square error (MSE) $\|\epsilon^{k-1}\|_2^2$. This is equivalent to finding the atom that is the most correlated to the residue ϵ^{k-1} (see Appendix B). In the shift-invariant case, the inner product between the residue and each atom ϕ_m is now replaced by the correlation with each kernel ψ_l (step 4), which is generally computed by fast Fourier transform. The noncircular complex correlation between signals $a(t)$ and $b(t)$ is given by

$$\Gamma\{a,b\}(\tau) = \langle a(t), b(t-\tau)\rangle = b^H(t-\tau)\, a(t). \tag{3}$$

The selection (step 6) determinates the optimal atom, characterized by its kernel index $l^k_{\max}$ and its position $\tau^k_{\max}$. An active dictionary D^k is formed, which collects all of the selected atoms (step 7), and the signal y is projected onto this selected subspace. Coding coefficients x^k are computed via the orthogonal projection of y on D^k (step 8). This is carried out recursively, by block matrix inversion [22]. The vector obtained, $x^k = [x_{l^1_{\max};\tau^1_{\max}}; x_{l^2_{\max},\tau^2_{\max}} \dots x_{l^k_{\max},\tau^k_{\max}}]^T$, is reduced to its active (i.e., nonzero) coefficients, denoting by $(\cdot)^T$, the transpose operator.

Different stopping criteria (step 11) can be used: a threshold on k, the number of iterations, a threshold on the relative root MSE (rRMSE) $\|\epsilon^k\|_2/\|y\|_2$, or a threshold on the decrease in the rRMSE. In the end, the OMP provides a K-sparse approximation of y:

$$\hat{y}^K = \sum_{k=1}^{K} x_{m^k_{\max}} \phi_{m^k_{\max}} = \sum_{k=1}^{K} x_{l^k_{\max},\tau^k_{\max}} \psi_{l^k_{\max}}\left(t-\tau^k_{\max}\right). \tag{4}$$

The convergence of OMP is demonstrated in [22], and its recovery properties are analyzed in [23] and [51].

Algorithm 1: $x = \mathbf{OMP}\ (y, \Psi)$

1: **initialization**: $k = 1, \epsilon^0 = y$, dictionary $D^0 = \varnothing$
2: **repeat**
3: **for** $l \leftarrow 1, L$ **do**
4: Correlation: $C^k_l(\tau) \leftarrow \Gamma\{\ \epsilon^{k-1},\ \psi_l\}(\tau)$
5: **end for**
6: Selection: $(l^k_{\max}, \tau^k_{\max}) \leftarrow \arg\max_{l,\tau} |C^k_l(\tau)|$
7: Active Dictionary: $D^k \leftarrow D^{k-1} \cup \psi_{l^k_{\max}}(t-\tau^k_{\max})$
8: Active Coefficients: $x^k \leftarrow \arg\min_x \|y - D^k x\|_2^2$
9: Residue: $\epsilon^k \leftarrow y - D^k x^k$
10: $k \leftarrow k+1$
11: **until** stopping criterion

B. Multivariate OMP Presentation

After the necessary OMP review, we now present the M-OMP (Algorithm 2), to handle the multivariate framework described previously (Sections III-A and III-C). The multivariate framework is mainly taken into account in the computation of the correlations (step 4) and the selection (step 6). The following notation is introduced: $a[u](t)$ is the uth component of the multivariate signal $a(t)$.

For a comparison between the two frameworks from Section III-A, we look at the selection step, with the objective function named as S:

$$\left(l^k_{\max}, \tau^k_{\max}\right) \leftarrow \arg\max_{l,\tau} S_l(\tau). \tag{5}$$

In the multichannel framework, selection is based on the interchannel energy:

$$S_l(\tau) = \sum_{u=1}^{V} |\Gamma\{\epsilon^{k-1}[u], \psi_l\}(\tau)|^s = \sum_{u=1}^{V} |\Gamma[u](\tau)|^s = \|\Gamma(\tau)\|_s^s \tag{6}$$

with $s = 1, 2$, or ∞ [38]. The search for the maximum of the ℓ_s norms applied to the vectors $\Gamma(\tau)$ is equivalent to applying a mixed norm to the correlation matrix Γ. This provides a structured sparsity that is similar to the Group-lasso, as explained in [40].

In the multivariate framework that we will consider, the selection is based on the average correlation of the V components. Using the definition of the inner product given in Section I, we have

$$\begin{aligned} S_l(\tau) &= \left|\sum_{u=1}^{V} \Gamma\{\epsilon^{k-1}[u], \psi_l[u]\}(\tau)\right| = \left|\sum_{u=1}^{V} \Gamma[u](\tau)\right| \\ &= |\mathrm{trace}(\Gamma(\tau))| = |\langle \epsilon^{k-1}(t), \psi_l(t-\tau)\rangle|. \end{aligned} \tag{7}$$

In fact, this selection is based on the inner product, which is comparable to the classical OMP, but in the multivariate case.

Added to the difference of the approximation models (Section III-A), these two frameworks do not select the same atoms. Due to the absolute values, the noncollinearity or anticollinearity of components $\Gamma[u]$ are not taken into account in (6), and the multichannel selection is only based on the energy. The multivariate selection (7) is more demanding, and it searches for the optimal tradeoff between components $\Gamma[u]$: it keeps the atom that best-fits the residue in each component. The selections are equivalent if all $\Gamma[u]$ are collinear and in

the same direction. The differences between these two types of selection have also been discussed in [41] and [52].

The active dictionary D^k is also multivariate (step 7). For the orthogonal projection (step 8), the multivariate signal $y \in \mathbb{C}^{N\times V}$ (respectively, dictionary $D^k \in \mathbb{C}^{N\times k\times V}$) is vertically unfolded along the dimension of the components in a unicomponent vector $y\rfloor \in \mathbb{C}^{NV\times 1}$ (respectively, matrix $D^k\rfloor \in \mathbb{C}^{NV\times k}$). Then, the orthogonal projection of $y\rfloor$ on $D\rfloor$ is recursively computed, as in the unicomponent case, using block matrix inversion [22]. For this step, in the multichannel framework coefficients x are simply computed via the orthogonal projection of y on the active dictionary D [31].

At the end of this, the M-OMP provides a multivariate K-sparse approximation of y. Compared to the OMP, the complexity of the M-OMP is only increased by a factor of V, the number of components.

Algorithm 2: $x = \mathbf{Multivariate\_OMP}\ (y, \Psi)$

1: **initialization**: $k = 1,\ \epsilon^0 = y$, dictionary $D^0 = \varnothing$
2: **repeat**
3: **for** $l \leftarrow 1, L$ **do**
4: Correlation: $C_l^k(\tau) \leftarrow \sum_{u=1}^{V} \Gamma\{\epsilon^{k-1}[u], \psi_l[u]\}(\tau)$
5: **end for**
6: Selection : $(l^k_{\max}, \tau^k_{\max}) \leftarrow \arg\max_{l,\tau} |C_l^k(\tau)|$
7: Active Dictionary: $D^k \leftarrow D^{k-1} \cup \psi_{l^k_{\max}}(t - \tau^k_{\max})$
8: Active Coefficients: $x^k \leftarrow \arg\min_x \|y\rfloor - D^k\rfloor x\|^2$
9: Residue: $\epsilon^k \leftarrow y - D^k x^k$
10: $k \leftarrow k + 1$
11: **until** stopping criterion

V. Multivariate Dictionary Learning Algorithm

In this section, we first provide a global presentation of the Multivariate DLA, and then remarks are given. Added to the multivariate aspect, the novelty of this DLA is to combine shift-invariance and online learning.

A. Algorithm Presentation

For more simplicity, a non-shift-invariant formalism is used in this short introduction, with the atoms dictionary Φ. We have a training set of multivariate signals $Y = \{y_p\}_{p=1}^P$ and the index p is added to the variables. In our learning algorithm, named M-DLA (Algorithm 3), each training signal y_p is treated one at a time. This is an *online* alternation between two steps: a multivariate sparse approximation and a multivariate dictionary update. The multivariate sparse approximation (step 4) is carried out by M-OMP:

$$x_p = \arg\min_x \|y_p - \Phi x\|^2 \text{ s.t.} \|x\|_0 \leq K \tag{8}$$

and the multivariate dictionary update (step 5) is based on maximum likelihood criterion [6], on the assumption of Gaussian noise

$$\Phi = \arg\min_{\Phi} \|y_p - \Phi x_p\|^2 \text{ s.t.} \quad \forall\, m \in \mathbb{N}_M, \|\phi_m\| = 1. \tag{9}$$

This criterion is usually optimized by gradient descent [12]–[14]. To achieve this optimization, we set up a stochastic Levenberg–Marquardt second-order gradient descent [53]. This increases the convergence speed, blending together the stochastic gradient and Gauss–Newton methods. The current iteration is denoted as i. For each multivariate kernel ψ_l, the update rule is given by (see Appendix B):

$$\psi_l^i(\underline{t}) = \psi_l^{i-1}(\underline{t}) + \left(H_l^i + \lambda^i \cdot I\right)^{-1} \cdot \sum_{\tau\in\sigma_l} x^{i*}_{l,\tau;p}\, \epsilon_p^{i-1}(\underline{t} + \tau), \tag{10}$$

with $\underline{t}$ as the indices limited to the ψ_l temporal support, λ the adaptive descent step, and H_l the Hessian computed as explained in Appendix C. This step is called LM-update (step 5). There are multiple strategies concerning the adaptive step: the classical choice $\lambda^i = \lambda_0 \cdot i$ is made (with $\lambda_0 = 1$). The multivariate framework is taken into account in the dictionary update, with all of the components $\psi_l[u]$ of the multivariate kernel ψ_l updated simultaneously. Moreover, the kernels are normalized at the end of each iteration, and their lengths can be modified. Kernels are lengthened if there is some energy in their edges and shortened otherwise.

At the beginning of the algorithm, the kernels initialization (step 1) is based as white Gaussian noise. At the end, different stopping criteria (step 8) can be used: a threshold on the rRMSE computed for the whole of the training set, or a threshold on i, the number of iterations. In M-DLA, the M-OMP is stopped by a threshold on the number of iterations. We cannot use rRMSE here, because at the beginning of the learning, the kernels of white noise cannot span a given part of the space studied.

Algorithm 3: $\Psi = \mathbf{Multivariate\_DLA}\ (\{y_p\}_{p=1}^P)$

1: **initialization**: $i = 1, \Psi^0 = \{L$ kernels of white noise$\}$
2: **repeat**
3: **for** $p \leftarrow 1, P$ **do**
4: Sparse Approximation: $x_p^i \leftarrow$ M$-$OMP (y_p, Ψ^{i-1})
5: Dictionary Update: $\Psi^i \leftarrow$ LM-update (y_p, x_p^i, Ψ^{i-1})
6: $i \leftarrow i + 1$
7: **end for**
8: **until** stopping criterion

B. Remarks About the Learning Processes

In this paragraph, a non-shift-invariant formalism is used for simplicity, with the atom dictionary Φ. We define $Y = \{y_p\}_{p=1}^P$ as the training set.

The learning algorithms K-SVD [8] and ILS-DLA [18] have *batch* alternation: sparse approximation is carried out for the whole finite training set Y, and then the dictionary is updated. If the usual convergence of the algorithms is observed empirically, theoretical proof of the strict decrease in the MSE at each iteration is not available, due to the nonconvexity of the sparse approximation step carried out using ℓ_0-Pursuit algorithms. Convergence properties for dictionary learning are discussed in [54] and [55].

An *online* (also known as *continuous* or *recursive*) alternation can be set up, where each training signal is processed one at a time. The dictionary is updated after the sparse approximation of each signal y_p (so there is P more updates than for batch alternation). The processing order of the training signals is often random, so as not to influence the optimization path in a deterministic way. The first-order stochastic gradient descent used in [11] provides a learning algorithm with low memory and computational requirements, with respect to batch algorithms.

Bottou and Bousquet [56] explained that in an iterative process, each step does not need to be minimized perfectly to reach the expected solution. Thus, they proposed the use of stochastic gradient methods. Based on this, the faster performances of online learning are shown in [57] and [58], for small and large datasets. An online alternation of ILS-DLA, known as recursive least-squares DLA (RLS-DLA), is presented in [59], and this also shows better performances. Our learning algorithm is an online alternation, and we can tolerate fluctuations in the MSE. The stochastic nature of the online optimization allows a local minimum to be drawn out. Contrary to the K-SVD and ILS-DLA, we have never observed that the learning gets stuck in a local minimum close to the initial dictionary.

The nonconvex optimization of the M-OMP, the alternating minimization and the stochastic nature of our online algorithm do not allow to ensure the convergence of the M-DLA towards the global minimum. However we find a dictionary, minimum local or global, which assures the decompositions sparsity.

VI. The 2D Rotation Invariant Case

Having presented the M-OMP and the M-DLA, these algorithms are now simply specified for the 2D rotation invariant case.

A. Method Presentation

To process bivariate real data, we specify the multivariate framework for the bivariate signals. The signal under study, $y \in \mathbb{R}^{N\times 2}$, is now considered. Equation (2) becomes

$$\left\{ \begin{array}{l} y[1](t) \\ y[2](t) \end{array} \right\} = \sum_{l=1}^{L} \sum_{\tau\in\sigma_l} x_{l,\tau} \left\{ \begin{array}{l} \psi_l[1](t-\tau) \\ \psi_l[2](t-\tau) \end{array} \right\} + \epsilon(t) \qquad (11)$$

with $\{\cdot\}$ representing the multivariate concatenation, not the vertical one. This case will be referred to as the *oriented* case in the following, as bivariate real kernels cannot rotate and are defined in a fixed orientation.

Studying bivariate data, such as 2D movements, we aspire to characterize them independently of their orientations. M-OMP is now specified for this particular 2DRI case. The rotation invariance implies the introduction of a rotation matrix $R \in \mathbb{R}^{2\times 2}$ of angle $\theta_{l,\tau}$ for each bivariate real atom $\psi_l(t-\tau)$. So (11) becomes

$$\left\{ \begin{array}{l} y[1](t) \\ y[2](t) \end{array} \right\} = \sum_{l=1}^{L} \sum_{\tau\in\sigma_l} x_{l,\tau} R(\theta_{l,\tau}) \left\{ \begin{array}{l} \psi_l[1](t-\tau) \\ \psi_l[2](t-\tau) \end{array} \right\} + \epsilon(t). \qquad (12)$$

Now, in the selection step (Algorithm 2, step 6), the aim is to find the angle $\theta_{l^k_{\max},\tau^k_{\max}}$ that maximizes the correlations $|C^k_l(\tau,\theta_{l,\tau})|$. A naive approach is the sampling of $\theta_{l,\tau}$ into Θ angles and the addition of a new degree of freedom in the correlations computation (Algorithm 2, step 4). The complexity is increased by a factor of Θ with respect to the M-OMP used in the oriented case. Note that this idea is used for processing bidimensionnal signals $y \in \mathbb{R}^{N_1\times N_2}$ such as images [60], although this represents a problem different from ours.

To avoid this additional cost, we transform the signal y from $\mathbb{R}^{N\times 2}$ to $\mathbb{C}^N$ (i.e., $y \leftarrow y[1]+y[2]i$, with the imaginary number i). The kernels and coding coefficients are now complex as well. Retrieving (2), the M-OMP is now applied. For the coding coefficients, the modulus gives the coefficient amplitude and the argument gives the rotation angle:

$$x_{l,\tau} = |x_{l,\tau}| \cdot e^{i\,\theta_{l,\tau}}. \qquad (13)$$

Finally, the decomposition of signal $y \in \mathbb{C}^N$ is given as

$$y(t) = \sum_{l=1}^{L} \sum_{\tau\in\sigma_l} |x_{l,\tau}| \cdot e^{i\theta_{l,\tau}} \cdot \psi_l(t-\tau) + \epsilon(t). \qquad (14)$$

Now the kernel can be rotated, as here kernels are no longer learned through a particular orientation, as in the previous approach as *oriented* (M-OMP with $V=2$ and $y \in \mathbb{R}^{N\times 2}$). Thus, the kernels are shift and rotation invariant, providing a *non-oriented* decomposition (M-OMP with $V=1$ and $y \in \mathbb{C}^N$).

This 2DRI specification of the sparse approximation (*respectively,* dictionary learning) algorithm is now denoted as 2DRI-OMP (*respectively,* 2DRI-DLA). It is important to note that the 2DRI implementations are not different from the algorithms presented before; they are just specifications. Only the initial arrangement of the data and the use of the argument of the coding coefficients are different.

B. Notes

In the multisensor case, V sensors that acquire bivariate signals are considered. The sensors are physically linked, and so they are under the same rotation. For example, bivariate real signals from a velocity sensor (for velocities v_x and v_y), an accelerometer (for accelerations a_x and a_y), a gyrometer (for angular velocities g_x and g_y), etc., can be studied. These signals can be aggregated together in $y \in \mathbb{C}^{N\times 3}$ such that

$$y = \left\{ \begin{array}{l} v_x + v_y i \\ a_x + a_y i \\ g_x + g_y i \end{array} \right\}. \qquad (15)$$

Here, the common rotation angle is jointly chosen between the three complex components due to the multivariate methods. Thus, when used with several complex components, M-OMP (respectively, M-DLA) can be viewed as a joint 2DRI-OMP (respectively, 2DRI-DLA).

We also note that when the number of active atoms $K=1$, the 2DRI problem considered is similar to 2D curve matching [61]. Schwartz and Sharir provided an analytic solution to compute $R(\theta_{l,\tau})$, although their approach is very long, as it is computed for each l and each τ. The use of the complex signals indicated above allows this problem to be solved nicely and cheaply.

Still considering $K=1$, Vlachos *et al.* [62] provided rotation invariant signatures for trajectory recognition. However, as with most of methods based on invariant descriptors, their method loses rotation parameters, which is contrary to our approach.

VII. Application Data and Experiments

After having defined our methods, we present in this section the data that are processed and then the experimental results.

A. Application Data

Our methods are applied to the *Character Trajectory* motion signals that are available from the University of California at Irvine (UCI) database [63]. They have been initially dealt with

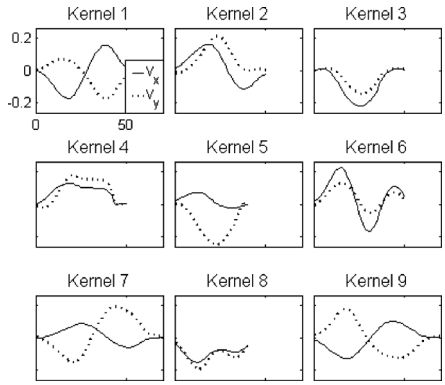


Fig. 2. Non-oriented learned dictionary (NOLD) of the velocities processed by 2DRI-DLA. Each kernel is composed of the real part v_x (solid line) and the imaginary part v_y (dotted line).

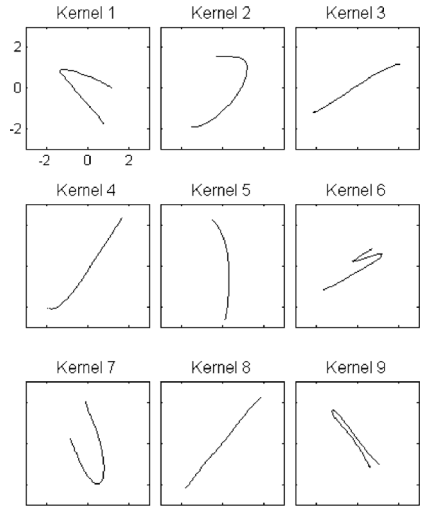


Fig. 3. Rotatable trajectory dictionary associated to the non-oriented learned dictionary (NOLD) processed by 2DRI-DLA.

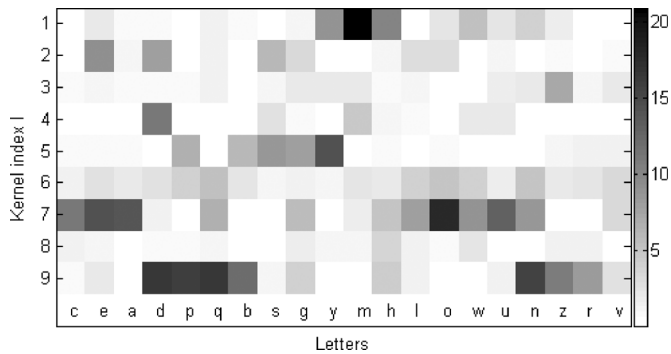


Fig. 4. Utilization matrix of the dictionary computed on the learning set. The means of the coefficient absolute values are given as a function of the kernel indices and the letters.

a probabilistic model and an expectation-maximization (EM) learning method [64], although without real sparsity in the resulting decompositions. The data comprise 2858 handwritten characters that were acquired with a Wacom tablet sampled at 200 Hz, with about a hundred occurrences of 20 letters written by the same person. The temporal signals are the Cartesian pen-tip velocities v_x and v_y. As the velocity units are not stated in the dataset description, we cannot define this here.

Using the raw data, we aim to learn an adapted dictionary to code the velocity signals sparsely. A partition of the database signals is made, as a training set for applying M-DLA, which is composed of 20 occurrences of each letter ($P = 400$ characters), and a test set for qualifying the sparse coding efficiency ($Q = 2458$ characters). These two sets are used in the following sections.

Although some of the comparisons are made with the oriented case, the results are mainly presented in the non-oriented case. For the differences in data arrangement, we note that in the oriented case, the signals are set as $y \leftarrow \{v_x;\ v_y\}^T$, whereas in the non-oriented case they are set as $y \leftarrow v_x + v_y i$. In these two cases, the dictionary learning algorithms begin their optimization with kernels initialized on white Gaussian noise.

Three experiments are now detailed, for the dictionary learning, the decompositions on the data, and the decompositions on the revolved data.

B. Experiment 1: Dictionary Learning

In this experiment, the 2DRI-DLA is going to provide a non-oriented learned dictionary (NOLD). The velocities are used to have the kernels as null at their edges. This avoids the introduction of discontinuities in the signal during the sparse approximation.[4] The kernel dictionary is initialized on white Gaussian noise, and 2DRI-DLA is applied to the training set. We obtain a velocity kernel dictionary as shown in Fig. 2, where each kernel is composed of the real part v_x (solid line) and the imaginary part v_y (dotted line). This convention for the line style in Fig. 2 will be used henceforth.

The velocity signals are integrated only to provide a more visual representation. However, due to the integration, the two different velocities kernels provide very similar trajectories (integrated kernels). The integrated kernel dictionary (Fig. 3) shows that motion primitives are successfully extracted by the 2DRI-DLA. Indeed, the straight and curved strokes shown in Fig. 3 correspond to the elementary patterns of the set of handwritten signals.

The question is how to choose the dictionary size hyperparameter L. In the non-oriented case, 9 kernels are used, whereas in the oriented case, 12 are required. The choice is an empirical tradeoff between the final rRMSE obtained on the training set, the sparsity of the dictionary, and the *interpretability* of the resulting dictionary (criteria that depend on the application can also be considered).

As interpretability is a subjective criterion, a utilization matrix is used in supervised cases (Fig. 4). The mean of the coefficients absolute values (gray shading level) computed on the learning set is mapped as a function of the kernel index l (ordinate), with the signal class as a letter (abscissa). The letters are organized according to the similarities of their utilization profiles. We can say that a dictionary has a good interpretability when well-used kernels are common to different letters that have related shapes (intuitively, other tools can be imagined to define a dictionary). For example, letters c, e and a have some similarities and share kernel 7. Similarly, d and p share kernel 9.

We also note here that during M-DLA, M-OMP provides a K-sparse approximation (Section V-A). K is the number of active coefficients, and it determinates the number of underlying primitives (atoms) that are searched and then learned from each

[4]Note also that contrary to the position signals, the velocity signals allow spatial invariance (different from the temporal shift-invariance).

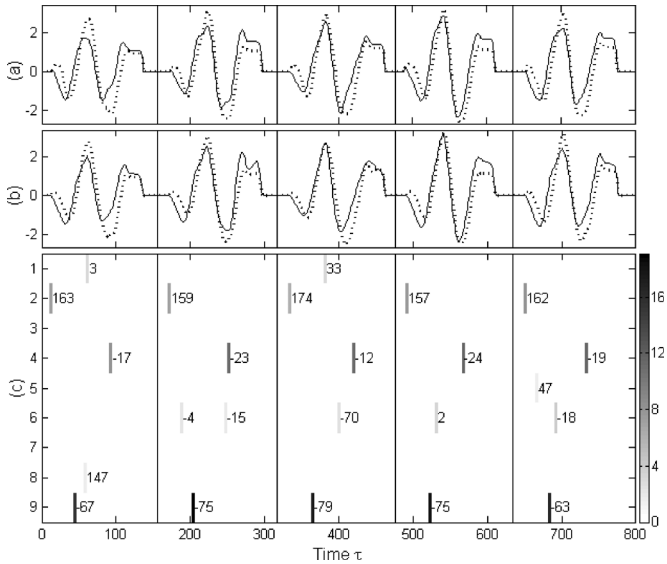


Fig. 5. Original (a) and reconstructed (b) velocity signals of five occurrences of the letter d (real part, solid line; imaginary part, dotted line), and their associated spikegram (c).

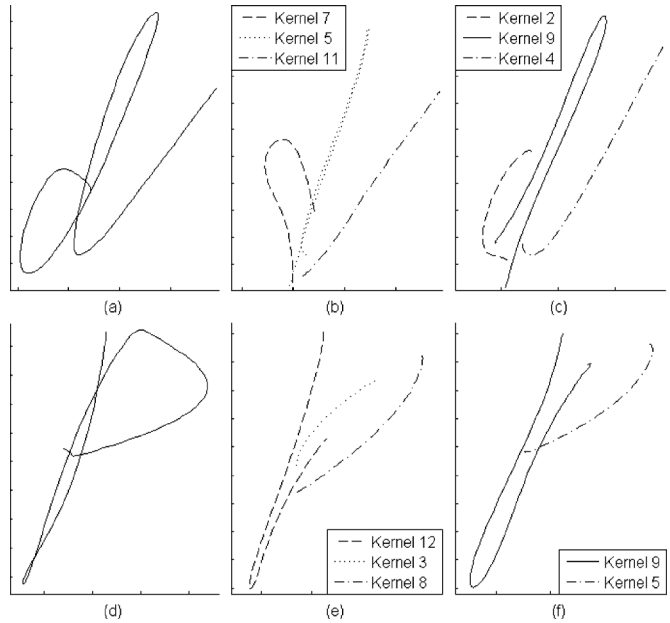


Fig. 6. Letter d (respectively, p). Original (a) [respectively, (d)], oriented reconstructed (b) [respectively, (e)], and non-oriented reconstructed (c) [respectively, (f)] trajectories.

signal. If the dictionary size L is too small compared to the ideal total number of primitives we are searching, the kernels will not be characteristic of any particular shape and the rRMSE will be high. Conversely, if L and K are particularly important, the dictionary learning will tend to scatter the information into the plentiful kernels. Here, the utilization matrix will be very smooth, without any kernel characteristics for particular letters. If L is particularly important and K is optimal, we can see that some kernels will be characteristic and well used, while others will not be. The utilization matrix rows of unused kernels are white, and it is easy to prune these to obtain the optimal dictionary. Typically, in our dictionary, kernel 8 can obviously be pruned (Fig. 4). Therefore, it is preferable to slightly overestimate L.

Finally, the crucial question is how to choose the parameter K. Indeed, this choice is empirical, as it depends on the number of primitives that the user forecasts to be in each signal of the dataset studied. In our experiment, we choose $K = 5$, as 2–3 primary primitives coding the main information, and the remaining ones coding the variabilities.

The nonconvex optimization of the M-OMP and the random processing of the training signals induce different dictionaries that are obtained with the same parameters. However, the variance of the results is small, and sometimes we obtain exactly the same dictionaries, or they have similar qualities (rRMSE, dictionary size, interpretability). For the following experiments, note that an oriented learned dictionary (OLD) is also processed by M-DLA.

C. Experiment 2: Decompositions on the Data

To evaluate the sparse coding qualities, non-oriented decompositions of five occurrences of the letter d on the NOLD are considered in Fig. 5. The velocities [Fig. 5(a)] [respectively, Fig. 5(b)] are the original (respectively, reconstructed, i.e., approximated) signals, which are composed of the real part v_x (solid line) and the imaginary part v_y (dotted line). The rRMSE on the velocities is around 12%, with 4–5 atoms used for the reconstruction (i.e., approximation). The coding coefficients $x_{l,\tau}$ are illustrated using a time-kernel representation [Fig. 5(c)] called spikegram [13]. This provides the four variables:

1) the temporal position τ (abscissa);
2) the kernel index l (ordinate);
3) the coefficient amplitude $|x_{l,\tau}|$ (gray shading level);
4) the rotation angle $\theta_{l,\tau}$ (number next to each spike, in degrees).

The low number of atoms used for the signal reconstruction shows the decomposition sparsity, which we refer to as the *sparse code*. The primary atoms are the largest amplitude ones, like kernels 2, 4, and 9, and these concentrate the relevant information. The secondary atoms code the variabilities between different realizations. The reproducibility of the decompositions is highlighted by the primary-atom repetition (amplitudes and angles) of the different occurrences. The sparsity and reproducibility are the proof of an adapted dictionary. Note that the spikegram is the result of the signal deconvolution through the learned dictionary.

The trajectory of the original letter d [Fig. 6(a)] [respectively, p, Fig. 6(d)] is reconstructed with the primary atoms. We compare the oriented case [Fig. 6(b)] [respectively, Fig. 6(e)] using the OLD and the non-oriented case [Fig. 6(c)] [respectively, Fig. 6(f)] using the NOLD. For instance, for the reconstruction, the letter d [Fig. 6(c)] is rebuilt as the sum of the NOLD kernels 2, 4, and 9 (the shapes can be seen in Fig. 3), which are specified by the amplitudes and the angles of the spikegram (Fig. 5(c)). We now focus on the principal vertical stroke that is common to letters d and p [Fig. 6(a) and (d)]. To code this, the oriented case uses two different kernels: kernel 5 for d [Fig. 6(b), dotted line] and kernel 12 for p [Fig. 6(e), dashed line]. However, the non-oriented case needs only one kernel for these two letters: kernel 9 [Fig. 6(c) and (f), solid line], which is used with an average rotation of 180°. Thus, the non-oriented approach reduces

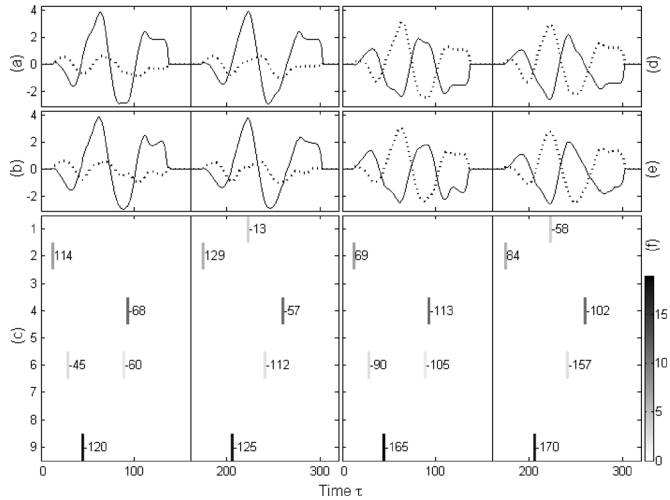


Fig. 7. Velocity signals revolved by $-45°$ (a) [respectively, $-90°$ (d)] and reconstructed (b) [respectively, (e)] for two occurrences of the letter d, and their associated spikegram (c) [respectively, (f)].

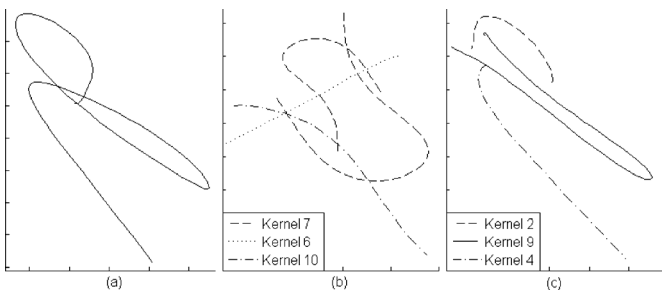


Fig. 8. Trajectory of letter d revolved by (a) an angle of $-90°$, and (b) the oriented reconstructed and (c) the non-oriented reconstructed.

the dictionary redundancy and provides an even more compact rotatable kernel dictionary. The detection of rotational invariants allows the dictionary size to decrease from 12 for the OLD, to 9 for the NOLD.

D. Experiment 3: Decompositions on Revolved Data

To simulate the rotation of the acquiring tablet, we artificially revolved the data of the test set, with the characters now rotated by angles of $-45°$ and $-90°$ (with the previous dictionaries kept). Fig. 7 shows the non-oriented decompositions of the second and third occurrences of the examples used in Fig. 5. The velocity signals rotated by $-45°$ [Fig. 7(a)] [respectively, $-90°$, Fig. 7(d)] are reconstructed in a non-oriented approach [Fig. 7(b)] [respectively, Fig. 7(e)]. In these two cases, the rRMSE is identical to the previous experiment, when the characters were not revolved. Fig. 7(c) [respectively, Fig. 7(f)] shows the associated spikegrams. The angle differences of the primary kernels between the spikegrams [Figs. 5(c), 7(c), and (f)] correspond to the angular perturbation we applied. This shows the rotation invariance of the decomposition.

The trajectory of letter d revolved by $-90°$ [Fig. 8(a)] is reconstructed with the primary kernels, with a comparison of the oriented case [Fig. 8(b)] using the OLD, and the non-oriented case [Fig. 8(c)] using the NOLD. In the oriented case, the rRMSE increases from 15% [Fig. 6(b)] to 30% [Fig. 8(b)], and the sparse coding is less efficient. Moreover, the selected kernels

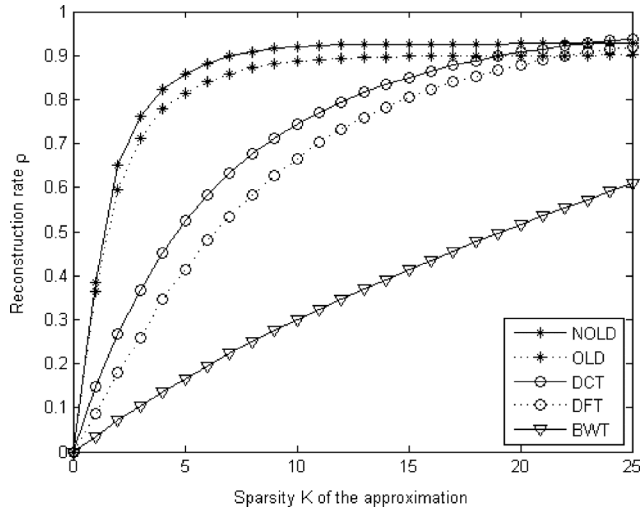


Fig. 9. Reconstruction rate ρ on the test set as a function of the sparsity K of the approximation for the different dictionaries.

are different, with there being no more reproducibility. The difference between these two reconstructions shows the necessity to be robust to rotations. In the non-oriented case, the rRMSE is equal to whatever the rotation angle is [Figs. 6(c) and 8(c)], and it is always less than the oriented case. The selected kernels are identical in the two cases, and they show the rotation invariance of the decomposition.

To conclude this section, the methods have been validated on bivariate signals and have shown rotation invariant sparse coding.

VIII. Comparisons

Three comparisons are made in this section: the dictionaries learned by our algorithms are first compared to classical dictionaries, then they are compared together, and finally the M-DLA is compared to the other dictionary learning algorithms.

A. Comparison With Classical Dictionaries

In this section, the test set is used for the comparison, although the characters are not rotated any more, and only component v_x is considered (to be in the real unicomponent case). We compare the previous learned dictionaries for the non-oriented approach (the NOLD, with $L = 9$) and the oriented approach (the OLD, with $L = 12$) to the classical dictionaries based on fast transforms, including: discrete Fourier transform (DFT), discrete cosine transform (DCT), and biorthogonal wavelet transform (BWT) (different types of wavelets that give similar performances; we only present the CDF 9/7). For each dictionary, K-sparse approximations are computed on the test set, and the reconstruction rate ρ is then computed. This is defined as:

$$\rho = 1 - \frac{1}{Q}\sum_{q=1}^{Q}\frac{\|\epsilon_q\|_2}{\|y_q\|_2}. \tag{16}$$

The rate ρ is represented as a function of K in Fig. 9.

We see that for a very few coefficients, the signals are reconstructed better with learned dictionaries (NOLD $L = 9$ and OLD $L = 12$) than with Fourier based dictionaries (DFT and

TABLE I
RECONSTRUCTION RATE RESULTS ON THE TEST SET

ρ (%)	Y_1	Y_2	Y_3	Y_4
NOLD $L=9$	85.8	85.6	85.9	85.0
OLD $L=12$	81.6	79.6	77.0	77.5
OLD $L=18$	83.0	81.4	79.98	78.9
OLD $L=24$	83.9	82.6	81.3	79.4
OLD $L=30$	84.8	83.5	82.8	80.7

Fig. 10. Utilization matrix for the OLD ($L = 12$) on set Y_2. The means of the absolute values of the coefficients is given as a function of the kernel indices and the letters. The letters with $'$ are those that are revolved.

DCT), which are themselves better than[5] wavelets (BWT). The results show the optimality of learned dictionaries over generic ones. If the dictionary learning is long compared to fast transforms, it is computed a single time for a given application. For the NOLD, only 7 atoms are needed to reach a rate of 90%, and the asymptote is at 93%. Furthermore, $\rho_{\mathrm{NOLD}} \geq \rho_{\mathrm{OLD}}$ whatever K. Rotation invariance is thus useful even without data rotation, as it provides a better fit of the variabilities between the different realizations.

Rates beyond $K = 25$ are not represented in Fig. 9, although the classical dictionaries can be seen to reach a reconstruction rate of 100%; they span all of the space, in contrast to learned dictionaries. This is because generic dictionaries are *bases of the space*, whereas learned dictionaries can be considered as a sort of *bases of the studied phenomenon*. In DLAs, the sparse approximation algorithm selects strong energy patterns, and these are then learned. So all of the signal parts that are never selected are never learned, which generally means the noise, although not always.

B. Comparison Between Oriented and Non-Oriented Learning

In Section VII-D, we only evaluated the rotation invariance of the decompositions with rotated data, and not the rotation invariance of the learning. The data in the test set were revolved, but not the data of the learning set. Here, we propose to study the rotation invariance of the whole learning method with rotated training signals.

[5]Note that this is due to the piecewise sinus aspect of the signals studied. This confirms that DCT appears to be the more adapted to motion data [65].

TABLE II
SIMILARITY CRITERION RESULTS ON THE TEST SET

c (%)	Y_2	Y_3	Y_4
NOLD $L=9$	100	100	100
OLD $L=12$	18.7	24.7	67.3
OLD $L=18$	14.1	17.2	60.6
OLD $L=24$	15.2	12.3	58.8
OLD $L=30$	6.3	9.0	57.8

In this comparison, learning and decompositions are carried out on datasets Y (including the training set and the test set), which are revolved at different angles. Y_1 contains the original data, Y_2 contains the original data and the data revolved by 120°, Y_3 contains the original data and the data revolved by 120° and 240°, and Y_4 contains the original data and the data revolved by random angles. The training sets allow the learning of different dictionaries: the NOLD with 9 kernels and the OLD with 12, 18, 24, and 30 kernels. The decompositions on the test sets give the reconstruction rates ρ, with $K = 5$.

Table I gives the results of the reconstruction rates according to the datasets (columns) and the dictionary type (rows). For the non-oriented learning, the results are similar, whatever the dataset. For oriented learnings, the approximation quality increases with the kernel number. The extra kernels can span the space better. However, even with 30 kernels, the OLD shows results worse than the NOLD with only 9 kernels. Moreover, the reconstruction rate decreases when number of different angles in the dataset increases, with revolved letters are considered as new letters.

These results only allow the approximation quality to be seen, and not the rotation invariance and the reproducibility of the decompositions. So, a similarity criterion is going to be set up, using the utilization matrix. As explained in Section VII-B, this matrix is formed by computing the means of coefficients absolute values of the test-set decompositions. As seen in Fig. 10, the values are given as a function of the kernel indices (ordinate) and the letters (abscissa). Fig. 10 shows the utilization matrix computed on Y_2 for the OLD, with $L = 12$. It can be seen that it is not the same kernels that are used to code a letter and its rotation, denoted by $(\cdot)'$. To evaluate this phenomenon, the similarity criterion c is defined as the mean of the normalized scalar products between the column of a letter and that of its rotation.

Table II summarizes the mean scalar product c given in the percentage according to the datasets (columns) and the dictionary type (rows). The criterion definition and the test-set design were chosen to give $c = 100\%$ in the reference non-oriented case. This remains at 100% whatever the dataset, which shows the rotation invariance. However, in reality, it is no use to carry out learning on the rotated data. As seen in Section VII-D, non-oriented learning on the original data is sufficient for an adapted dictionary that is robust to rotations.

For the oriented learnings, although bigger dictionaries give better reconstruction rates (Table I), they have poorer similarity criteria, as multiple kernels tend to scatter the information.

Therefore, artificially increasing the dictionary size is not a good idea for sparse coding, because it damages the results. Furthermore, increasing the number of different angles in the dataset gives better reproducibility, as the signals no longer influence the learning through a fixed orientation, and consequently the oriented kernels are the more general.

C. Comparison With Other Dictionary Learning Algorithms

We now compare our method to other DLAs. The advantages of online learning have already been pointed out in [11], [57], [58], so our experiment is on the robustness to shift-invariance. M-DLA is used in real and unicomponent cases, to compare it with the existing learning methods: K-SVD [8], the shift-invariant version of K-SVD [46] known as SI-K-SVD, and the shift-invariant ILS-DLA [48] (the shift factor is set as up to 1), which is indicated as SI-ILS-DLA in the following.

This comparison is based on the experience described in [46]. A dictionary Ψ of $L = 45$ kernels is created randomly and the kernel length is $T = 18$ samples. The training set is composed of $P = 2000$ signals of length $N = 20$, and it is synthetically generated from this dictionary. For the kernels, circular shifts are not allowed, and so only three shifts are possible. Each training signal is composed of the sum of three atoms, for which the amplitudes, kernels indices and shift parameters are randomly drawn. White Gaussian noise is also added at several levels: an SNR of 10, 20, and 30 dB, and without noise. All of the learning algorithms are applied with the same parameters, with the dictionary initialization made on the training set, and the sparse approximation step carried out by OMP. The learned dictionary $\tilde{\Psi}$ is returned after 80 iterations. Classical K-SVD is also tested, with hopes of recovering an atoms dictionary of 135 atoms (the 45 kernels in the three possible shifts).
In the experiment, a learned kernel $\hat{\psi}_l$ is considered as detected, i.e., recovered, if its inner product μ_l with its corresponding original kernel ψ_l is such that

$$\mu_l = |\langle \psi_l, \hat{\psi}_l \rangle| \geq 0.99. \tag{17}$$

The high threshold of 0.99 was chosen by [46]. For each learning algorithm, the detection rate of the kernels is plotted as a function of the noise level, which was averaged over five tests (Fig. 11).

This experiment only tests the algorithm precision. In our case, the online alternation provides learning that is fast, but not so precise, due to the stochastic noise that is induced by the random choice of a training signal at each iteration. We observe that 80% of the $\{\mu_l\}_{l=1}^{L}$ are between 0.97 and 1.00, with only a few above the severe threshold of 0.99. To be comparable with batch algorithms, which are more precise at each step, the classical strategy for the adaptive step proposed in Section V-A is adapted to the constraints of this experiment. With 2000 training signals, we prefer to keep a constant step for one loop of the training set. Moreover, the step is increased faster, to provide satisfactory convergence after 80 iterations. For the first 40 iterations, the step is set up as: $\lambda^i = (i - p + 1)^{1.5}$, and then it is kept constant for the last iterations: $\lambda^i = 40^{1.5}$. The results obtained are now plotted in Fig. 11.

Fig. 11 shows that having a shift-invariant model is obviously relevant. For shift-invariant DLAs, this underlines their ability to recover the underlying shift-invariant features. However, we observe that the M-DLA performance decreases when the noise levels increase, contrary to SI-K-SVD and SI-ILS-DLA, which appear not to be influenced in this way. Despite its stochastic update, our algorithm recovers the original dictionary in a similar proportion to the batch algorithms. This experiment supports the analysis of [56] relating to learning, where each step does not need to be minimized exactly to converge towards the expected solution.

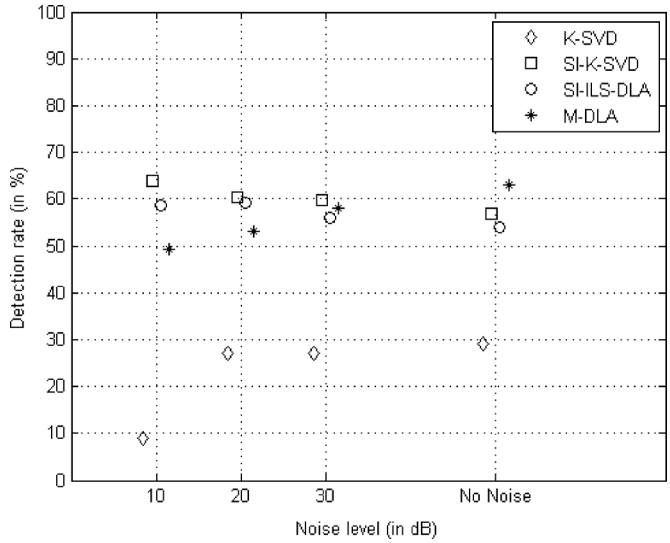


Fig. 11. Detection rate as a function of noise level for K-SVD (diamonds), SI-K-SVD (squares), SI-ILS-DLA (circles) and M-DLA (stars).

IX. Discussion

Dictionary learning allows signal primitives to be recovered. The resulting dictionary can be thought of as a catalog of elementary patterns that are dedicated to the application considered and that have a physical meaning, as opposed to classical dictionaries such as wavelets, curvelets, etc. Therefore, decompositions based on such a dictionary are made sparsely on the underlying features of the signal set studied. For the rRMSE, the few atoms used in the decompositions shows the efficiency of this sparse-coding method.

The non-oriented approach for sparse coding reduces the dictionary size in two ways:

1) when the signals studied cannot rotate, the non-oriented approach detects rotational invariants (the vertical strokes of letters d and p, for example), which reduces the dictionary size;
2) when the signals studied can rotate. To provide efficient sparse coding, the oriented approach needs to learn motion primitives for each of the possible angles. Conversely, in the non-oriented case, single learning is sufficient. This provides a noticeable reduction of the dictionary size.

In this way, the shift-invariant and rotation invariant cases provide a compact learned dictionary Ψ. Moreover, the non-oriented approach allows robustness for any writing direction (tablet rotation) and for any writing inclination (intra- and inter-user variabilities). When added to a classification step, the angles information allows the orientation of the writing baseline to be given.

Recently, Mallat notes [66] that the key for the classification is not the representation sparsity but its invariances. In our 2DRI case, the decompositions are invariant to temporal

shift (parameter τ), to rotation (parameter $\theta_{l,\tau}$), to scale (parameter $|x_{l,\tau}|$) and to spatial translation (use of velocity signals instead of position signals). Based on these considerations, we are also working on the classification of sparse codes, to carry out gesture recognition, and the first experiments look promising. Spikegrams appear to be good representations for classification, and their reproducibility can be exploited. The classification results are interesting, because kernels are learned only with ℓ_2 data-fitting criterion of unsupervised dictionary learning, and so without discriminative constraints. It appears that recovering the primitives underlying the features of a signal set via a sparsity constraint allows this set to be described discriminatively.

Motion data is new with regards to custom sparse coding applications. Recently, we have taken cognizance of a work made on multicomponent motion signals. In [67], Kim *et al.* use a tensor factorization with tensor constraints to make a multicomponent dictionary learning. Modelized by the multivariate framework and processed by our proposed algorithms, this problem is solved without the heavy tensor formalism.

X. Conclusion

In contrast to the well-known multichannel framework, a multivariate framework was introduced to more easily present our methods relating to bivariate signals. First, the multivariate sparse-coding methods were presented: Multivariate OMP, which provides sparse approximations for multivariate signals, and Multivariate DLA, which is a learning algorithm that empirically learns the optimal dictionary associated to a multivariate signal set. All of the dictionary components are updated simultaneously. The resulting dictionary jointly provides sparse approximations of all of the signals of the set considered. This DLA is an online alternation between a sparse approximation step carried out by M-OMP, and a dictionary update that is optimized by stochastic Levenberg–Marquardt second-order gradient descent. The online learning does not disturb the performance of the dictionary obtained, even in the shift-invariant case.

Then in dealing with bivariate signals, we wanted the decompositions to be independent of the orientation of the movement execution in 2D space. To provide rotation invariant sparse coding, the methods were simply specified to the 2D rotation invariant case, known as 2DRI-OMP and 2DRI-DLA. Rotation invariance is useful, but not only when the data are rotated, as it allows to code variabilities. Moreover, shift-invariant and rotation invariant cases induce a compact learned dictionary and are useful for classification. As validation, these methods were applied to 2D handwritten data.

The methods applications are dimensionality reduction, denoising, gesture representation and analysis, and all of the other processing that is based on multivariate feature extraction. The prospects under consideration are to extend these methods to 3D rotation invariance for trivariate signals, and to present the classification step that is applied to the spikegrams and the associated results.

Appendix A
Considerations for the Implementation

We are going to look at the OMP complexity for the different approaches in the shift-invariant case. Often enough, the acquired signals are dyadic (i.e., the signal size N is a power of 2). If they are not, they are lengthened by zero-padding to $\lceil N \rceil$ samples, with $\lceil N \rceil$ as the first power of 2 to the N. So, in the unicomponent case, the correlation is computed by FFT in $O(\lceil N \rceil \log \lceil N \rceil)$ for each kernel. In the multivariate case, the multivariate correlation is the sum of the V component correlations, and it is computed in $O(V \cdot \lceil N \rceil \log \lceil N \rceil)$ for each kernel.

To retrieve the classical case, we can simply vectorize the signal from $N \times V$ to $NV \times 1$. However, zero-padding between the components is necessary, otherwise the kernel components can overlap two consecutive signal components during the correlation. Limiting the kernels length to N_L samples (which is a loss of flexibility) with N_L the size of the longest kernel, zero-padding of N_L samples has to be carried out between two consecutive components.

This zero-padded signal of $V(N+N_L)$ samples is lengthened again, in order to be dyadic. Finally, the correlation complexity is $O((\lceil V(N+N_L) \rceil) \log(\lceil V(N+N_L) \rceil))$ for each kernel. Moreover, for the selection step, investigations need to be limited to the first $N+N_L$ samples of the correlation obtained. To conclude, the multivariate framework is easier to implement and has lower complexity than the classical framework with vectorized data.

Appendix B
Complex Gradient Operator

The gradient operator was introduced by Brandwood in [68]. Assuming $z \in \mathbb{C}$, the complex derivation rules are

$$\partial z^*/\partial z = \partial z/\partial z^* = 0 \quad \text{and} \quad \partial z/\partial z = \partial z^*/\partial z^* = 1.$$

[68] showed that the direction of maximum rate of change of an objective function $J = \|\epsilon\|_2^2$ with z is $\partial J/\partial z^*$:

$$\partial J/\partial z^* = \partial(\epsilon^H \epsilon)/\partial z^* = \epsilon^H \partial \epsilon/\partial z^* + \partial \epsilon^H/\partial z^* \epsilon.$$

1) The derivation of J with respect to x_m:

$$\begin{aligned} \partial \epsilon/\partial x_m^* &= \partial(y - \Phi x)/\partial x_m^* = 0, \\ \partial \epsilon^H/\partial x_m^* &= \partial(y^H - x^H \Phi^H)/\partial x_m^* = -\phi_m^H. \end{aligned}$$

Thus: $-\partial J/\partial x_m^* = \phi_m^H \epsilon = \langle \epsilon, \phi_m \rangle$. This gives the selection step of the OMP (algorithm 1).

2) The derivation of J with respect to ϕ_m:

$$\begin{aligned} \partial \epsilon/\partial \phi_m^* &= \partial(y - \Phi x)/\partial \phi_m^* = 0, \\ \partial \epsilon^H/\partial \phi_m^* &= \partial(y^H - x^H \Phi^H)/\partial \phi_m^* = -x_m^*. \end{aligned}$$

Thus: $-\partial J/\partial \phi_m^* = x_m^* \epsilon$. This gives the first-order part of the update of the M-DLA. The complex least mean squares (CLMS) obtained by the pseudo-gradient [69] is retrieved (give or take a factor of 2). For the complex Hessian, we make reference to [70].

In the shift-invariant case, all of the translations of a considered kernel ψ_l are taken into account in the dictionary update: $-\partial J/\partial \psi_l^* = \sum_{\tau \in \sigma_l} x_{l,\tau}^* \underline{\epsilon}_\tau$, with $\underline{\epsilon}_\tau$ the error localized at τ and restrained to the ψ_l temporal support (i.e., $\underline{\epsilon}_\tau = \epsilon|_{t=\tau \ldots \tau+T_l}$). This gives the shift-invariant update of the M-DLA (10).

APPENDIX C
CALCULUS OF THE HESSIAN

In this Appendix, we explain the calculation of the Hessian H_l. This allows the adaptive step to be specified to each kernel ψ_l, and the convergence of the well-used kernels to be stabilized at the beginning of the learning.

An average Hessian H_l is computed for each kernel ψ_l, not for each sample, to avoid fluctuations between neighboring samples. H_l is thus reduced to a scalar. Assuming the hypothesis of sparsity (a few atoms are used for the approximation), the overlap of selected atoms is initially considered as nonexistent. So the cross-derivative terms of H_l are null, and we have

$$H_l^i = \sum_{\tau \in \sigma_l} \left| x_{l,\tau}^i \right|^2 . \tag{18}$$

For overlapping atoms, the learning method can become unbalanced, due to the error in the gradient estimation. We overestimate the Hessian H_l slightly to compensate for this. All $\tau \in \sigma_l$ are sorted and then indexed by j, such that: $\tau_1 < \tau_2 \ldots < \tau_j < \tau_{j+1} \ldots < \tau_{|\sigma_l|}$, with $|\sigma_l|$ as the cardinal of the set σ_l. Denoting T_l^i as the length of the kernel ψ_l at the iteration i, the set J_l is defined as: $J_l = \{j \in \mathbb{N}_{|\sigma_l - 1|} : \tau_{j+1} - \tau_j < T_l^i\}$. This allows for the identification of overlap situations. The cross-derivative terms of H_l are no longer considered to be null, and their contributions are proportional to $x_{l,\tau_j}^{i*} x_{l,\tau_{j+1}}^i + x_{l,\tau_j}^i x_{l,\tau_{j+1}}^{i*} = 2\Re(x_{l,\tau_j}^{i*} x_{l,\tau_{j+1}}^i)$. Double-overlap situations are not considered, when $\tau_{j+2} - \tau_j < T_l^i$. Due to the hypothesis of sparsity, these situations are considered to be very rare (as is verified in practice), and they are compensated for by overestimating H_l. The absolute value of the cross terms is taken: $|x_{l,\tau_j}^{i*} x_{l,\tau_{j+1}}^i| \geq 2\Re(x_{l,\tau_j}^{i*} x_{l,\tau_{j+1}}^i)$. The absolute value is not disturbing, even without double overlap, as it is better to slightly overestimate H_l than to underestimate it (it would better move a little but surely). Finally, we propose for H_l the following approximation quickly computed:

$$H_l^i = \sum_{\tau \in \sigma_l} \left| x_{l,\tau}^i \right|^2 + 2 \sum_{j \in J_l} \frac{T_l^i - (\tau_{j+1} - \tau_j)}{T_l^i} \left| x_{l,\tau_j}^{i*} x_{l,\tau_{j+1}}^i \right| . \tag{19}$$

The following comments can be made regarding (19):

- if the gap between two atoms is always greater than T_l, the first approximation of (18) is recovered;
- when the overlap is weak, the cross-products have little influence on the Hessian;
- intra-kernel overlaps have been considered, but not inter-kernel ones. However, we see that inter-kernel overlaps do not disturb the learning, so we ignore their influence.

The update step based on (10) and (19) is called LM-update (step 5, Algorithm 3).

Without the Hessian in (10), a first-order update is retrieved. In this case, the convergence speed of a kernel is directly linked to the sum of its decomposition coefficients. Advantage of the Hessian is to tend to make the convergence speed similar for all kernels, independently of their uses in the decompositions. Concerning the approximation of the Hessian, at the beginning of the learning, kernels which are still white noises overlap frequently and method can become unbalanced. Increasing the Hessian, the approximation thus stabilizes the beginning of the learning process. After, since kernels converge, overlaps are quite rare and the approximation of the Hessian is closed to (18).

ACKNOWLEDGMENT

The authors would like to thank Z. Kowalski, A. Hanssen, and anonymous reviewers for their fruitful comments, and C. Berrie for his help about English usage.

REFERENCES

[1] A. Bruckstein, D. Donoho, and M. Elad, "From sparse solutions of systems of equations to sparse modeling of signals and images," *SIAM Rev.*, vol. 51, pp. 34–81, 2009.

[2] S. Mallat, *A Wavelet Tour of Signal Processing*, 3rd ed. New York: Academic, 2009.

[3] E. Candès and D. Donoho, "Curvelets—A surprisingly effective nonadaptive representation for objects with edges," in *Curves and Surfaces*. Nashville, TN: Vanderbilt Univ. Press, 2000, pp. 105–120.

[4] S. Chen, D. Donoho, and M. Saunders, "Atomic decomposition by basis pursuit," *SIAM J. Sci. Comput.*, vol. 20, pp. 33–61, 1998.

[5] J. Starck, M. Elad, and D. Donoho, "Redundant multiscale transforms and their application for morphological component analysis," *Adv. Imag. Electron Phys.*, vol. 132, pp. 287–348, 2004.

[6] B. Olshausen and D. Field, "Sparse coding with an overcomplete basis set: A strategy employed by V1?," *Vision Res.*, vol. 37, pp. 3311–3325, 1997.

[7] K. Kreutz-Delgado, J. Murray, B. Rao, K. Engan, T. Lee, and T. Sejnowski, "Dictionary learning algorithms for sparse representation," *Neural Comput.*, vol. 15, pp. 349–396, 2003.

[8] M. Aharon, M. Elad, and A. Bruckstein, "K-SVD: An algorithm for designing overcomplete dictionaries for sparse representation," *IEEE Trans. Signal Process.*, vol. 54, no. 11, pp. 4311–4322, 2006.

[9] M. Elad and M. Aharon, "Image denoising via sparse and redundant representations over learned dictionaries," *IEEE Trans. Image Process.*, vol. 15, no. 12, pp. 3736–3745, 2006.

[10] J. Mairal, M. Elad, and G. Sapiro, "Sparse representation for color image restoration," *IEEE Trans. Image Process.*, vol. 17, no. 1, pp. 53–69, 2008.

[11] M. Aharon and M. Elad, "Sparse and redundant modeling of image content using an image-signature-dictionary," *SIAM J. Imag. Sci.*, vol. 1, pp. 228–247, 2008.

[12] M. Lewicki and T. Sejnowski, "Coding time-varying signals using sparse, shift-invariant representations," in *Proc. Conf. Adv. Neural Inf. Process. Syst. II*, 1999, vol. 11, pp. 730–736.

[13] E. Smith and M. Lewicki, "Efficient coding of time-relative structure using spikes," *Neural Comput.*, vol. 17, pp. 19–45, 2005.

[14] T. Blumensath and M. Davies, "Sparse and shift-invariant representations of music," *IEEE Trans. Audio, Speech, Lang. Process.*, vol. 14, no. 1, pp. 50–57, 2006.

[15] B. Olshausen, "Sparse codes and spikes," in *Probabilistic Models of the Brain: Perception and Neural Function*. Cambridge, MA: MIT Press, 2001, pp. 257–272.

[16] G. Monaci, P. Vandergheynst, and F. Sommer, "Learning bimodal structure in audio-visual data," *IEEE Trans. Neural Netw.*, vol. 20, no. 12, pp. 1898–1910, 2009.

[17] K. Engan, S. Aase, and J. Husøy, "Multi-frame compression: Theory and design," *Signal Process.*, vol. 80, pp. 2121–2140, 2000.

[18] K. Engan, K. Skretting, and J. Husøy, "Family of iterative LS-based dictionary learning algorithms, ILS-DLA, for sparse signal representation," *Digit. Signal Process.*, vol. 17, pp. 32–49, 2007.

[19] G. Davis, "Adaptive Nonlinear Approximations," Ph.D. dissertation, New York Univ., New York, 1994.

[20] J. Tropp and S. Wright, "Computational methods for sparse solution of linear inverse problems," *Proc. IEEE*, vol. 98, pp. 948–958, 2010.

[21] S. Mallat and Z. Zhang, "Matching pursuits with time-frequency dictionaries," *IEEE Trans. Signal Process.*, vol. 41, no. 12, pp. 3397–3415, 1993.

[22] Y. Pati, R. Rezaiifar, and P. Krishnaprasad, "Orthogonal matching pursuit: Recursive function approximation with applications to wavelet decomposition," in *Proc. Asilomar Conf. Signals, Syst., Comput.*, Pacific Grove, CA, 1993, vol. 1, pp. 40–44.

[23] J. Tropp, "Greed is good: Algorithmic results for sparse approximation," *IEEE Trans. Inf. Theory*, vol. 50, no. 10, pp. 2231–2242, 2004.

[24] M. Osborne, B. Presnell, and B. Turlach, "A new approach to variable selection in least squares problems," *IMA J. Numer. Anal.*, vol. 20, pp. 389–404, 2000.

[25] I. Daubechies, M. Defrise, and C. De Mol, "An iterative algorithm for linear inverse problems with a sparsity constraint," *Commun. Pure Appl. Math.*, vol. LVII, pp. 1413–1457, 2004.

[26] I. Tosic and P. Frossard, "Dictionary learning," *IEEE Signal Process. Mag.*, vol. 28, pp. 27–38, 2011.
[27] M. Yaghoobi, T. Blumensath, and M. Davies, "Dictionary learning for sparse approximations with the majorization method," *IEEE Trans. Signal Process.*, vol. 57, no. 6, pp. 2178–2191, 2009.
[28] R. Gribonval, "Sparse decomposition of stereo signals with matching pursuit and application to blind separation of more than two sources from a stereo mixture," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process. (ICASSP)*, Orlando, FL, 2002.
[29] R. Gribonval, "Piecewise linear source separation," in *Proc. SPIE*, 2003, vol. 5207, pp. 297–310.
[30] S. Lesage, S. Krstulović, and R. Gribonval, "Under-determined source separation: Comparison of two approaches based on sparse decompositions," in *Proc. Int. Workshop Independent Compon. Anal. Blind Signal Separation*, 2006, pp. 633–640.
[31] R. Gribonval, H. Rauhut, K. Schnass, and P. Vandergheynst, "Atoms of all channels, unite! Average case analysis of multi-channel sparse recovery using greedy algorithms," IRISA, Rennes, France, Tech. Rep. PI-1848, 2007.
[32] A. Lutoborski and V. Temlyakov, "Vector Greedy algorithms," *J. Complex.*, vol. 19, pp. 458–473, 2003.
[33] D. Leviathan and V. Temlyakov, "Simultaneous approximation by greedy algorithms," Univ. of South Carolina, Columbia, SC, Tech. Rep., 2003.
[34] D. Leviathan and V. Temlyakov, "Simultaneous greedy approximation in Banach spaces," *J. Complex.*, vol. 21, pp. 275–293, 2005.
[35] J. Tropp, A. Gilbert, and M. Strauss, "Algorithms for simultaneous sparse approximation; Part I: Greedy pursuit," *Signal Process.—Sparse Approximations in Signal Image Process.*, vol. 86, pp. 572–588, 2006.
[36] J. Tropp, "Algorithms for simultaneous sparse approximation; Part II: Convex relaxation," in *Signal Process.—Sparse Approximations in Signal Image Process.*, 2006, vol. 86, pp. 589–602.
[37] S. Cotter, B. Rao, K. Engan, and K. Kreutz-Delgado, "Sparse solutions to linear inverse problems with multiple measurement vectors," *IEEE Trans. Signal Process.*, vol. 53, no. 7, pp. 2477–2488, 2005.
[38] J. Chen and X. Huo, "Theoretical results on sparse representations of multiple-measurement vectors," *IEEE Trans. Signal Process.*, vol. 54, no. 12, pp. 4634–4643, 2006.
[39] D. Baron, M. Duarte, S. Sarvotham, M. Wakin, and R. Baraniuk, "An information-theoretic approach to distributed compressed sensing," in *Proc. Allerton Conf. Commun., Control, Comput.*, Allerton, IL, 2005.
[40] A. Rakotomamonjy, "Surveying and comparing simultaneous sparse approximation (or group-Lasso) algorithms," *Signal Process.*, vol. 91, pp. 1505–1526, 2011.
[41] R. Gribonval and M. Nielsen, "Beyond sparsity: Recovering structured representations by ℓ_1-minimization and greedy algorithms—Application to the analysis of sparse underdetermined ICA," IRISA, Rennes, France, Tech. Rep. PI-1684, 2005.
[42] B. Mailhé, R. Gribonval, F. Bimbot, and P. Vandergheynst, "A low complexity orthogonal matching pursuit for sparse signal approximation with shift-invariant dictionaries," in *Proc. IEEE Conf. Acoust., Speech, Signal Process. (ICASSP)*, Taipei, Taiwan, 2009, pp. 3445–3448.
[43] M. Mørup, M. Schmidt, and L. Hansen, "Shift invariant sparse coding of image and music data," Technical Univ. of Denmark, Lyngby, Denmark, Tech. Rep., 2008.
[44] H. Wersing, J. Eggert, and E. Körner, "Sparse coding with invariance constraints," in *Proc. Int. Conf. Art. Neural Netw. (ICANN)*, 2003, pp. 385–392.
[45] P. Jost, P. Vandergheynst, S. Lesage, and R. Gribonval, "MoTIF: An efficient algorithm for learning translation invariant dictionaries," in *Proc. IEEE Conf. Acoust., Speech, Signal Process. (ICASSP)*, Toulouse, France, 2006, vol. 5, pp. 857–860.
[46] M. Aharon, "Overcomplete dictionaries for sparse representation of signals," Ph.D. dissertation, Technion—Israel Inst. of Technol., Haifa, Israel, 2006.
[47] B. Mailhé, S. Lesage, R. Gribonval, F. Bimbot, and P. Vandergheynst, "Shift-invariant dictionary learning for sparse representations: Extending K-SVD," in *Proc. Eur. Signal Process. Conf. (EUSIPCO)*, Lausanne, Switzerland, 2008.
[48] K. Skretting, J. Husøy, and S. Aase, "General design algorithm for sparse frame expansions," *Signal Process.*, vol. 86, pp. 117–126, 2006.
[49] Q. Barthélemy, A. Larue, A. Mayoue, D. Mercier, and J. Mars, "Multivariate dictionary learning and shift & 2D rotation invariant sparse coding," in *Proc. IEEE Workshop Stat. Signal Process. (SSP)*, Nice, France, 2011, pp. 645–648.
[50] Q. Barthélemy, A. Larue, A. Mayoue, D. Mercier, and J. Mars, "Apprentissage de dictionnaires multivariés et décomposition parcimonieuse invariante par translation et par rotation 2D," in *Proc. XXIII Colloque GRETSI—Traitement du Signal et des Images* (in French), Bordeaux, France, 2011.
[51] R. De Vore and V. Temlyakov, "Some remarks on greedy algorithms," *Adv. Comput. Math.*, vol. 5, pp. 173–187, 1996.
[52] P. Durka, A. Matysiaka, E. Montes, P. Sosa, and K. Blinowskaa, "Multichannel matching pursuit and EEG inverse solutions," *J. Neurosci. Methods*, vol. 148, pp. 49–59, 2005.
[53] K. Madsen, H. Nielsen, and O. Tingleff, "Methods for non-linear least squares problems," Technical Univ. of Denmark, Lyngby, Denmark, Tech. Rep., 2004, 2nd ed.
[54] M. Aharon, M. Elad, and A. Bruckstein, "On the uniqueness of overcomplete dictionaries, and a practical way to retrieve them," *Linear Algebra Its Appl.*, vol. 416, pp. 48–67, 2006.
[55] R. Gribonval and K. Schnass, "Dictionary identification—Sparse matrix-factorization via ℓ_1-minimization," *IEEE Trans. Inf. Theory*, vol. 56, no. 7, pp. 3523–3539, 2010.
[56] L. Bottou and O. Bousquet, "The tradeoffs of large scale learning," *Adv. Neural Inf. Process. Syst.*, vol. 20, pp. 161–168, 2008.
[57] J. Mairal, F. Bach, J. Ponce, and G. Sapiro, "Online dictionary learning for sparse coding," in *Proc. Int. Conf. Mach. Learn. (ICML)*, Montreal, QC, Canada, 2009.
[58] J. Mairal, F. Bach, J. Ponce, and G. Sapiro, "Online learning for matrix factorization and sparse coding," *J. Mach. Learn. Res.*, vol. 11, pp. 19–60, 2010.
[59] K. Skretting and K. Engan, "Recursive least squares dictionary learning algorithm," *IEEE Trans. Signal Process.*, vol. 58, no. 4, pp. 2121–2130, 2010.
[60] M. Mørup and M. Schmidt, "Transformation invariant sparse coding," in *Proc. Mach. Learn. Signal Process. (MLSP)*, Beijing, China, 2011.
[61] J. Schwartz and M. Sharir, "Identification of partially obscured objects in two and three dimensions by matching noisy characteristic curves," Courant Inst. of Math. Sci., New York Univ., New York, Tech. Rep. Robotics Rep. 46, 1985.
[62] M. Vlachos, D. Gunopulos, and G. Das, "Rotation invariant distance measures for trajectories," in *Proc. SIGKDD Int. Conf. Knowledge Discovery Data Mining*, Seattle, WA, 2004, pp. 707–712.
[63] A. Frank and A. Asuncion, "UCI machine learning repository," 2010 [Online]. Available: http://archive.ics.uci.edu/ml
[64] B. Williams, M. Toussaint, and A. Storkey, "A primitive based generative model to infer timing information in unpartitioned handwriting data," in *Proc. Int. Joint Conf. Art. Intell. (IJCAI)*, Hyderabad, India, 2007, pp. 1119–1124.
[65] I. Akhter, Y. Sheikh, S. Khan, and T. Kanade, "Trajectory space: A dual representation for nonrigid structure from motion," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 7, pp. 1442–1456, 2011.
[66] S. Mallat, "Group invariant scattering," CMAP, Palaiseau, France, Tech. Rep., 2011.
[67] T. Kim, G. Shakhnarovich, and R. Urtasun, "Sparse coding for learning interpretable spatio-temporal primitives," in *Proc. Neural Inf. Process. Syst. (NIPS)*, Vancouver, Canada, 2010.
[68] D. Brandwood, "A complex gradient operator and its application in adaptive array theory," in *Proc. Inst. Electr. Eng. F—Commun., Radar, Signal Process.*, 1983, vol. 130, pp. 11–16.
[69] B. Widrow, J. McCool, and M. Ball, "The complex LMS algorithm," *Proc. IEEE*, vol. 63, pp. 719–720, 1975.
[70] A. van den Bos, "Complex gradient and Hessian," *Proc. Inst. Electr. Eng.—Vision, Image, Signal Process.*, vol. 141, pp. 380–383, 1994.

Quentin Barthélemy received the Engineering degree and the M.Res. in signal and images analysis and processing (with distinction) from Grenoble Institut National Polytechnique (Grenoble INP), France, both in 2009.

Currently, he is working towards the Ph.D. degree in signal processing at the CEA-LIST (Alternative Energies and Atomic Energy Commission), France, from 2010. His research interests include sparse approximation and dictionary learning specified to the shift and rotation invariance cases, and their applications to multivariate signals.

Anthony Larue received the Aggregation in electrical engineering at the Ecole Normale Suprieure de Cachan, France, in 2002, the M.S. degree in automatic control and signal processing from the University of Paris XI, France, in 2003, and the Ph.D. degree in signal processing from Institut National Polytechnique of Grenoble, France, in 2006. His Ph.D. dissertation deals with blind deconvolution of noisy data for seismic applications.

He joined the CEA-LIST, Gif-sur-Yvette, France, in 2006. His research interests are signal processing, machine learning, and especially sparse decomposition of multicomponent signals. Since 2010, he has been the head of the Data Analysis Tools Laboratory, which developed data analysis algorithms for health, security or energy applications.

Aurélien Mayoue received the Engineering degree from Institut National Polytechnique of Grenoble (INPG), France, in 2005.

He spent one year at Ecole Polytechnique Fédérale of Lausanne (EPFL) as an exchange student. He worked in the biometrics field at TELECOM Sud-Paris, France, in 2006 and joined the CEA-LIST, Gif-sur-Yvette, France, in 2009, where he is a Research Engineer in signal and image processing. He is involved in MotionLab (collaborative lab between MOVEA and CEA) dedicated to the creation of innovative motion-aware applications for mass-market products. His interests are sparse coding and motion data processing.

David Mercier received the Engineering degree in computer science from the Ecole Supérieure d'Electricité (SUPELEC) in 1999 and the Ph.D. degree in signal processing from Rennes 1 University in 2003.

He followed with a postdoctoral position at the Detection and Geophysics Laboratory of the CEA. His work focused on signal processing and machine learning tools for seismic applications like event or wave classification. In 2005, he joined the CEA-LIST, Gif-sur-Yvette, France, and he is currently head of the Information, Models and Learning Laboratory from 2010. He is interested in decision-making systems. His research activities cover data analysis, signal processing and machine learning in several fields like geophysics, industrial monitoring, energy markets, genetics, and crisis conduct.

Jérôme I. Mars (M'08) received the M.S. degree in geophysics from Joseph Fourier University of Grenoble, France, in 1986 and the Ph.D. degree in signal processing from the Institut National Polytechnique of Grenoble, France, in 1988.

From 1989 to 1992, he was a Postdoctoral Researcher at the Centre des Phénomènes Alatoires et Geophysiques, Grenoble, France. From 1992 to 1995, he was a Visiting Lecturer and Scientist at the Materials Sciences and Mineral Engineering Department at the University of California, Berkeley. He is currently Professor in Signal Processing for the Department Image and Signal at GIPSA-Lab (UMR 5216), Grenoble Institute of Technology, France, where he is head of the Signal Image Department. His research interests include seismic and acoustic signal processing, source separation methods, and time frequency time-scale characterization.

Dr. Mars is member of EAGE.